OPOS Quickstart

Introduction

(Taken from https://en.wikipedia.org/wiki/OPOS)

OPOS, full name OLE for Retail POS, a platform specific implementation of UnifiedPOS, is a point of sale device standard for Microsoft Windows operating systems that was initiated by Microsoft, NCR, Epson, and Fujitsu-ICL and is managed by the Association for Retail Technology Standards. The OPOS API was first published in January 1996. The standard uses component object model and, because of that, all languages that support COM controls (i.e. Visual C++, Visual Basic, and C#) can be used to write applications.

The OPOS standard specifies two levels for an OPOS control, the control object which presents an abstract hardware interface to a family of devices such as receipt printer and the service object which handles the interface between the control object and the actual physical device such as a specific model of receipt printer. This division of functionality provides a way for the application development to write to an abstract hardware interface while allowing the application to work with a variety of different hardware. The only requirement is that a hardware vendor supplies an OPOS compatible service object with their particular hardware offering.

Typically a manufacturer of point of sale terminals will provide along with a terminal operating system an OPOS control object package with a software utility that is used to configure OPOS settings. Such a utility will specify the settings for an OPOS control object and indicate the service object to be used with a particular OPOS profile. When the point of sale application starts up, it loads the OPOS control object and the OPOS control object in turn loads the service object specified by the current OPOS profile. The Windows Registry is typically used as the persistent store for device settings. The hardware device manufacturer will normally provide a utility for device specific settings used by the service object.

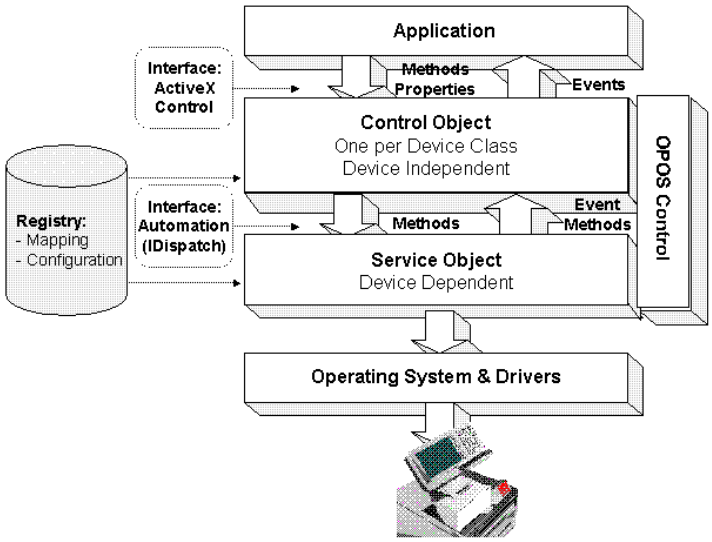
OPOS Model

(Taken from http://monroecs.com/oposbackground.htm)

OLE for Retail POS Controls adhere to the ActiveX Control specifications. They expose properties, methods, and events to a containing Application. The controls are invisible at run time, and rely exclusively upon the containing application for requests through methods and sometimes properties. Responses are given to the application through method return values and parameters, properties, and events.

The OLE for Retail POS software is implemented using the layers shown in the following

diagram:

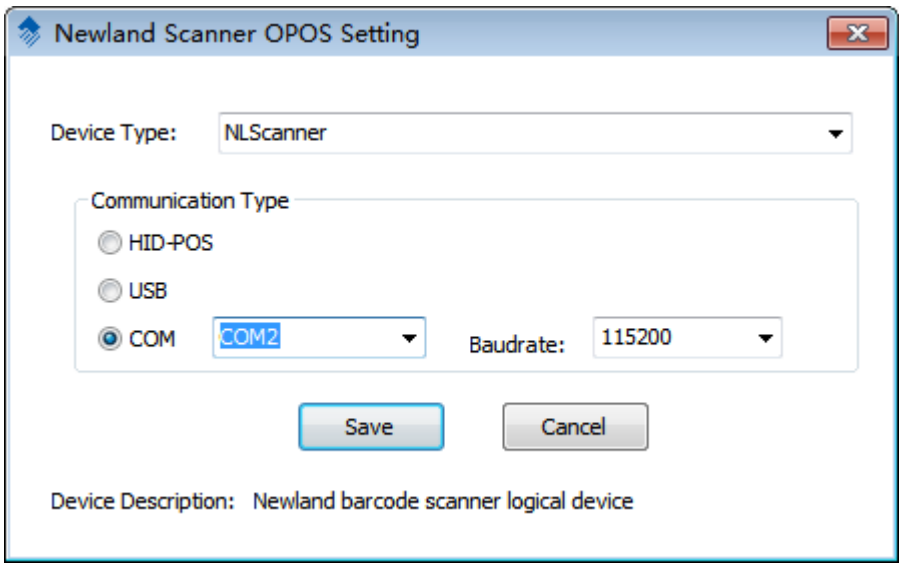


Information For User

● Usage

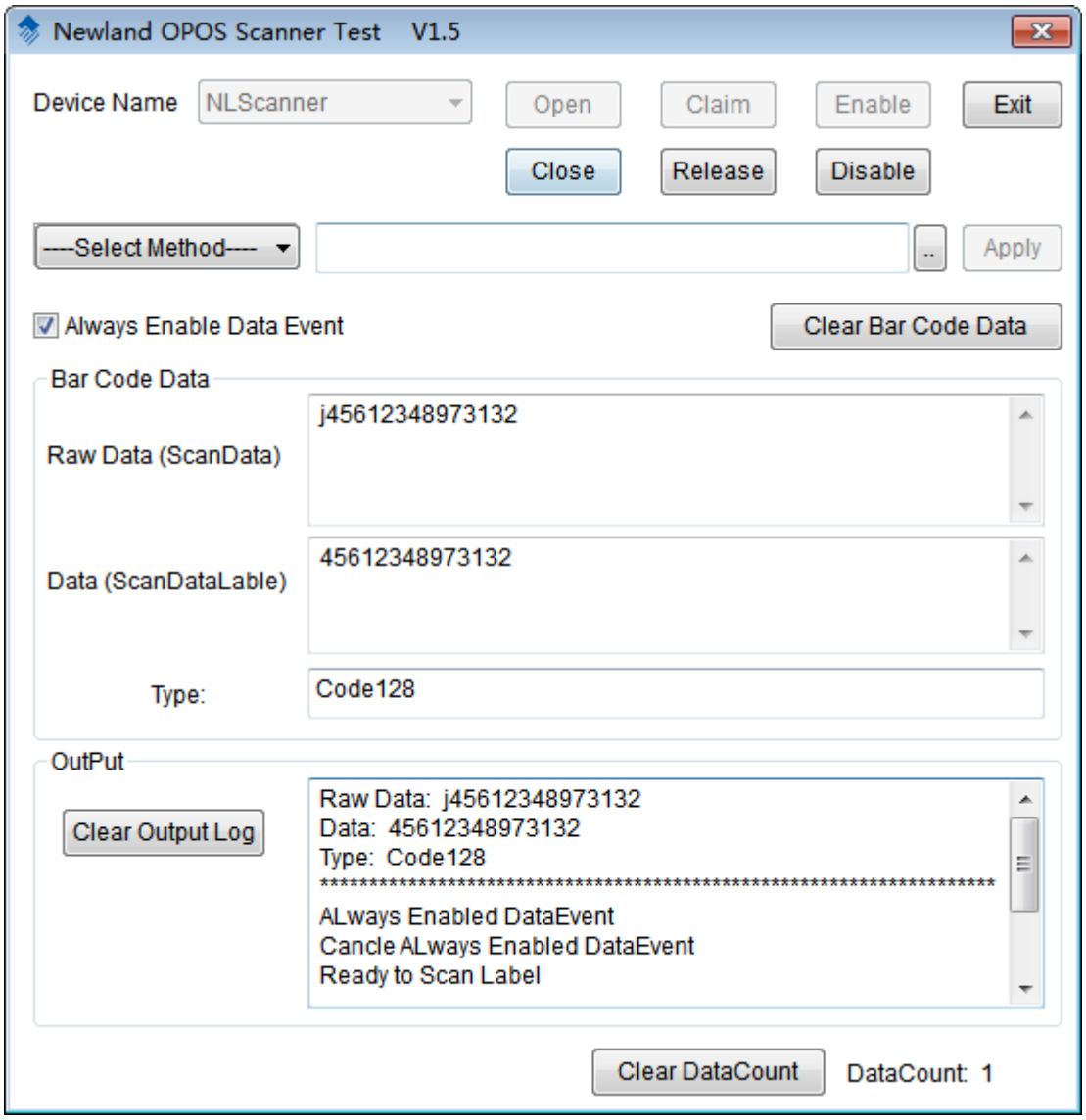
1. Install "OPOS Newland Scanner V1.1.0.msi". No need to install "OPOS CCO Runtime", as the necessary file "OPOSSoScanner.dll" is included in the installer.
2. Switch to the install directory.
3. Change the interface settings:

Run "Newland Scanner OPOS Setting.exe" under the "Scanner OPOS Configuration Utility" folder.



4. Run Test Program:
   Run "OPOSScannerTest.exe" under the "Newland Scanner OPOS" folder.
   Click Open, Claim, Enable Button, then scan a barcode.

Button Name	Notes
Open	
Close	
Claim	Acquire exclusive access to the device
Release	Remove exclusive access to the device
Enable	Enable the device
Disable	Disable the device
Always Enable DataEvent:	By default, DataEvent is auto disable after a data incoming.



Information For Developers

● Registry Settings

64-bit system:
[HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\OLEForRetail\ServiceOPOS\Scanner\NLScanner]

32-bit system:
[HKEY_LOCAL_MACHINE\SOFTWARE\OLEForRetail\ServiceOPOS\Scanner\NLScanner]

<table>
<tr><td>Value</td><td>Type</td><td>Notes</td></tr>
<tr><td>Baudrate</td><td>REG_SZ</td><td></td></tr>
<tr><td>DeviceType</td><td>REG_DWORD</td><td>Must be 3</td></tr>
<tr><td>CommType</td><td>DeviceType</td><td>The value is determined by the scanner device’s interface:

<table><tr><td>Interface</td><td>Value</td></tr><tr><td>RS232</td><td rowspan="2">COMxx (xx is the serial port number)
For example: COM2</td></tr><tr><td>USB COM</td></tr><tr><td>USB DataPipe</td><td>USB</td></tr><tr><td>HID POS</td><td>HID-POS</td></tr></table></td></tr>
</table>

● OPOS Scanner Properties:

Name	Notes
AutoDisable	
BinaryConversion	Not support binary conversion
CapCompareFirmwareVersion	
CapPowerReporting	
CapStatisticsReporting	
CapUpdateFirmware	
CapUpdateStatistics	
CheckHealthText	Not implemented
Claimed	
DataCount	

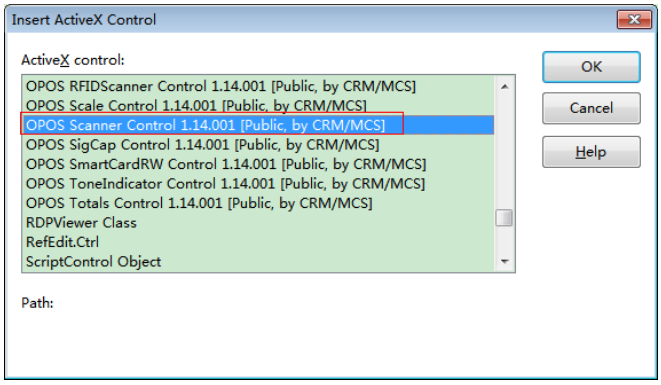
DataEventEnabled	
DeviceEnabled	
FreezeEvents	
OpenResult	
OutputID	Not support asynchronous output
PowerNotify	Not support
PowerState	Not support
ResultCode	
ResultCodeExtended	
State	
ControlObjectDescription	
ControlObjectVersion	
ServiceObjectDescription	
ServiceObjectVersion	
DeviceDescription	
DeviceName	
DecodeData	
ScanData	
ScanDataLabel	
ScanDataType	

- **OPOS Scanner Methods**

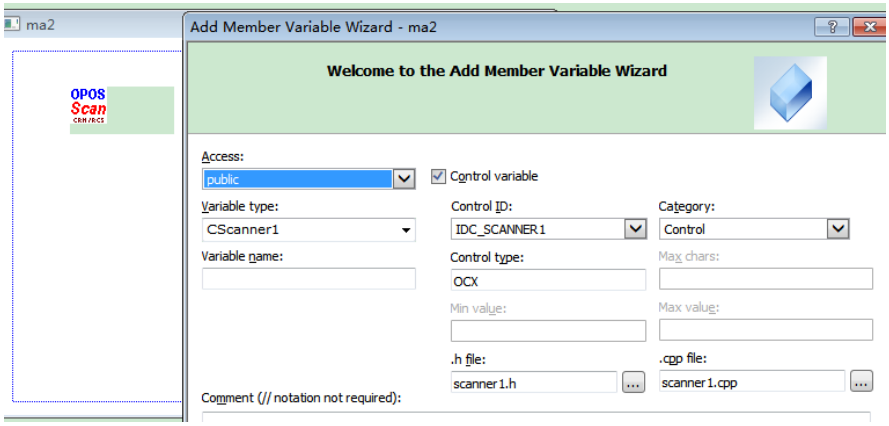
Name	Notes
Open	
Close	
Claim	
Release	
CheckHealth	Not implemented
ClearInput	
ClearInputProperties	Not implemented
ClearOutput	Not implemented
DirectIO	
CompareFirmwareVersion	
ResetStatistics	Not implemented
RetrieveStatistics	Not full implemented
UpdateFirmware	Not implemented
UpdateStatistics	Not implemented

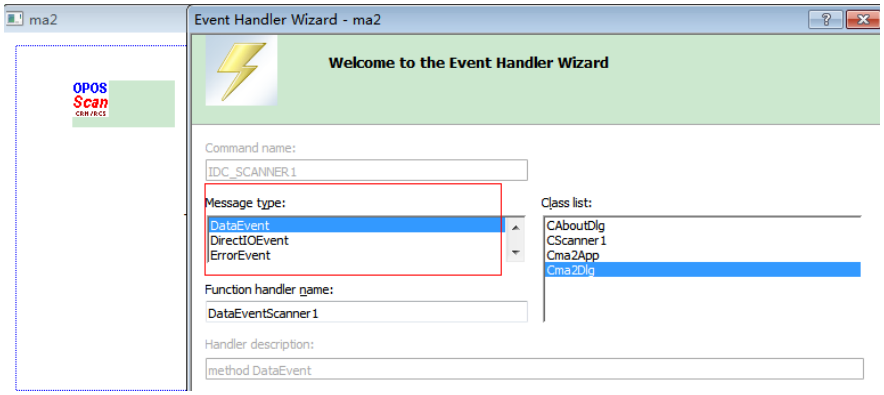
Use Control

1. Start Visual Studio 2008 and create a MFC Application.
2. Switch to resource editor.
3. Right click and select "Insert ActiveX Control...".



4. Choose an item starts with "Opos Scanner Control", and click ok.
5. Right click the added window, create variable with ID "IDC_SCANNER1", and add event handler.





Sample Code

```
// Define a member variable for the ActiveX control:  m_scanner
long result = 0;
unsigned timeout = 500;  // 500 ms
do {
     // logicalName may be  "NLScanner"
     result = m_scanner.Open(logicalName);
     if (result != OPOS_SUCCESS) break;

     // wait maximum timeout to get exclusive access to the device.
     result = m_scanner.ClaimDevice(timeout);
     if (result != OPOS_SUCCESS) break;

     // enable device
     m_scanner.SetDeviceEnabled(true);

     //To receive scanned barcode events
     m_scanner.SentDataEventEnabled (true);


     // deal with data event
     // ...


     // remove exclusive access
     result = m_scanner.ReleaseDevice();
     if (result != OPOS_SUCCESS) break;

     result = m_scanner.close();
     if (result != OPOS_SUCCESS) break;

} while(0);

show_error(result);
```

Useful Link

Document download: http://monroecs.com/unifiedpos.htm
Introduction: http://monroecs.com/oposccos.htm